1. **Introdução**

Quando estamos desenvolvendo uma aplicação baseada na arquitetura cliente-servidor, a comunicação acontece através de requisições e respostas. Por exemplo, um cliente (ex: navegador) envia uma requisição e o servidor (local onde está nossa aplicação) retorna uma resposta. Até o momento não há muita novidade nessas afirmações. No entanto, essa aula tem como objetivo compreender melhor o que são requisições assíncronas e como usá-las.

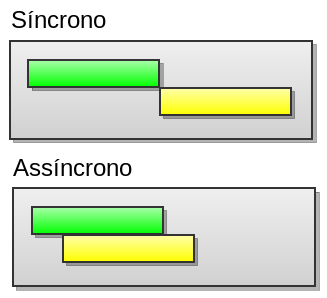
1. **Revisão: Tipos de requisição**

2.1- Síncrona

Quando uma requisição é enviada, o processo remetente é bloqueado até que ocorra uma resposta, ou seja, não é possível enviar novas requisições até que nossa requisição atual seja finalizada, pois existe sincronismo entre as requisições.

2.2- Assíncrona

Em uma requisição assíncrona, não existe sincronismo entre as requisições. Assim, podemos enviar diversas requisições em paralelo e cada resposta retorna quando estiver pronta.



1. **Exemplo de requisição assíncrona**

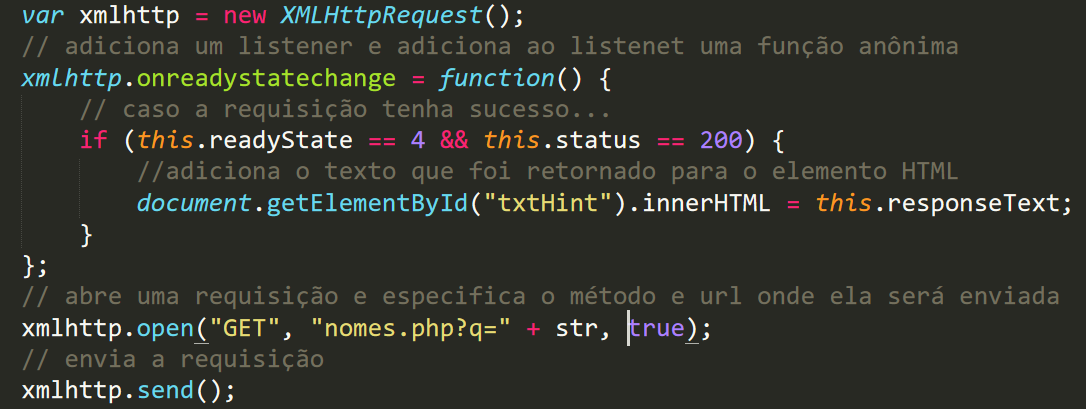
Imagine que você é o cozinheiro da noite e precisa fazer um jantar para sua família. Você sabe que o jantar precisa ser servido às 19h, logo, você planeja toda cozinha para isso. Imagine que você agora tem a opção de fazer todo o jantar de forma síncrona e assíncrona, qual você escolheria?

Se você escolher de forma síncrona, você primeiro deve descongelar o frango e aguardar essa tarefa terminar. Quando o frango estiver pronto você o tempera e começa a cozinhá-lo. Somente após o término do cozimento você poderá fazer outra tarefa. Por outro lado, se você escolher fazer de forma assíncrona, você poderá cozinhar o arroz enquanto o frango cozinha ou até mesmo fazer uma salada enquanto o frango descongela.

1. **Como fazer uma requisição assíncrona com o Javascript?**

Para realizar uma requisição assíncrona é necessário utilizar um objeto específico do Javascript que permite que a requisição seja enviada. Esse objeto é o XMLHttpRequest().

Observe a imagem abaixo:



Após instanciado, é necessário adicionar um *listener* que lidará com o resultado da requisição. Lembrando que, como a requisição é assíncrona, simplesmente instanciar o objeto não resolve o problema. É necessário que seja indicada a URL que será feita o *request* e logo após executar o método *send*().